
**THE CONDENSED GUIDE
TO SOFTWARE ACQUISITION
BEST PRACTICES**



For additional information please contact the
Software Program Managers Network

(703) 521-5231 • Fax (703) 521-2603

E-Mail: spmn@aol.com

<http://www.spmn.com>



June 1998

WHY THIS GUIDE?



This publication was prepared for the

Software Program Managers Network
4600 North Fairfax Drive, Suite 302
Arlington, VA 22203

The ideas and findings in this publication should not be construed as an official DoD position. It is published in the interest of scientific and technical information exchange.

Norm Brown
Director, Software Program Managers Network

Copyright © 1998 by Computers & Concepts Associates

This work was created by Computers & Concepts Associates in the performance of Space and Naval Warfare Systems Command (SPAWAR) Contract Number N00039-94-C-0153 for the operation of the Software Program Managers Network (SPMN).

As software has become increasingly critical to defense systems, DoD's ability to effectively manage the development and maintenance of software has not kept pace. This guide provides winning strategies used by successful government and industry software program managers—practices and tools that, if utilized, will enable the effective management of large-scale software programs.

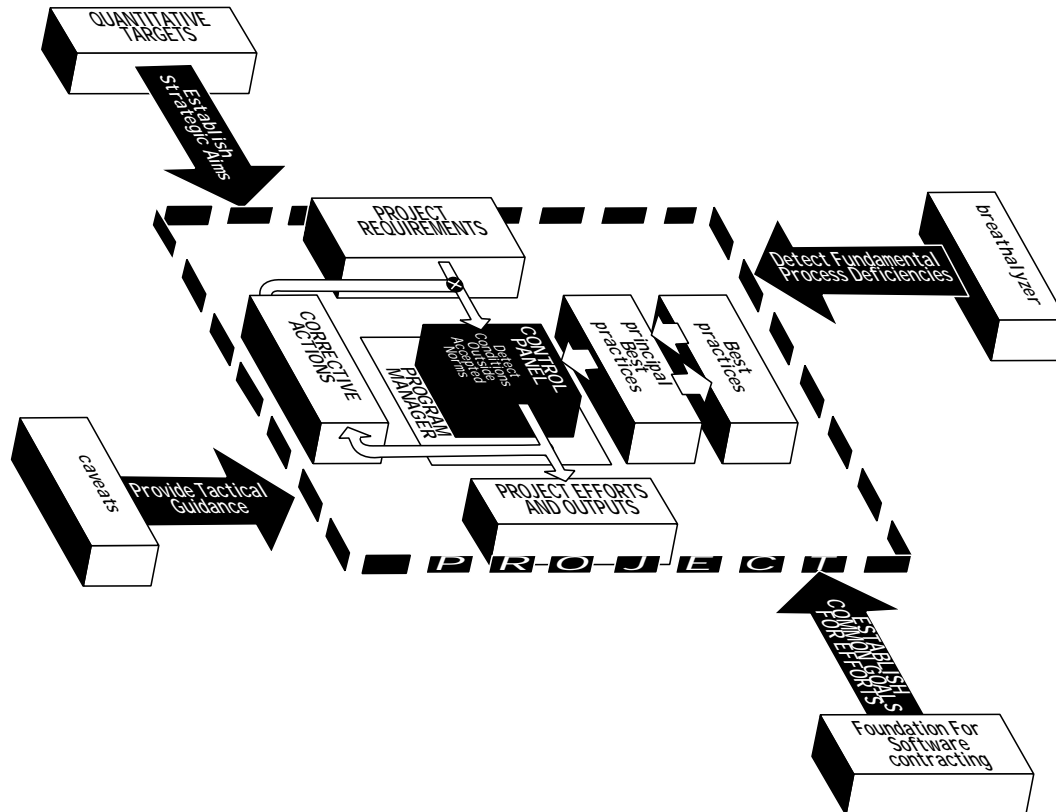
- **Project Control Panel** enables project status to be monitored.
- **“Breathalyzer” Test** helps determine if a project is fit to be “on the road.”
- **Quantitative Targets** provide best-in-class objectives and associated warning levels of possible malpractice.
- **Principal Best Practices** are essential to the success of any large-scale software project.
- **Best Practices** are used successfully in industry and government and are recommended for consideration.
- **Project Caveats** are “worst practices” to avoid.

Readers desiring more detailed information should refer to the expanded version of this guide, *The Program Manager's Guide to Software Acquisition Best Practices*.

We would appreciate any comments or suggestions you have (e-mail: spmnaol.com).

Norm Brown
Executive Director

1. HOW THE PIECES FIT TOGETHER

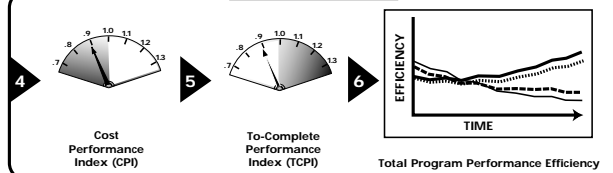
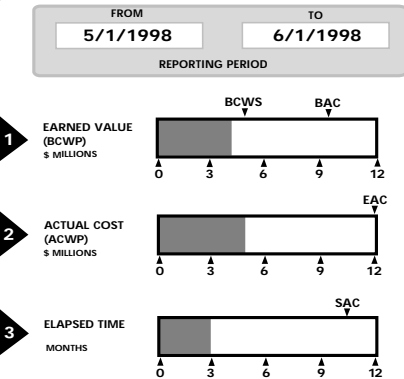


2. PROJECT CONTROL PANEL

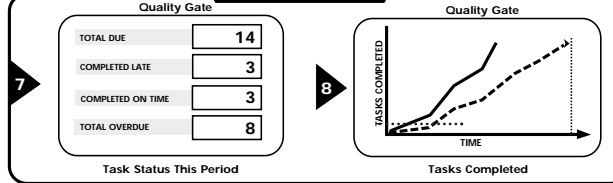
PROJECT CONTROL PANEL

PROGRESS

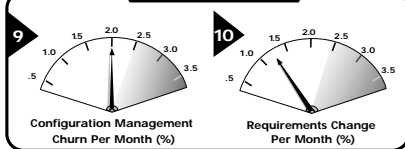
PRODUCTIVITY



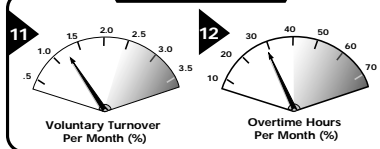
COMPLETION



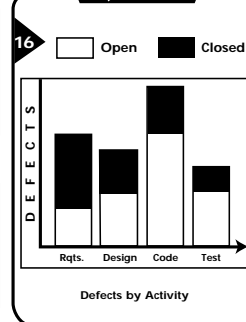
CHANGE



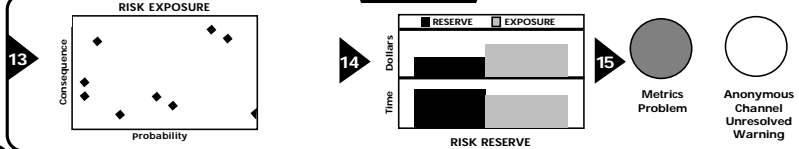
STAFF



QUALITY



RISK



- ▶ Cumulative earned value (BCWP) delivered compared with total budgeted cost (BAC) and cumulative planned value (BCWS).
- ▶ Cumulative actual cost (ACWP) compared with total estimated cost at completion (EAC).
- ▶ Current reporting time period compared with total periods budgeted.
- ▶ $CPI = \frac{BCWP}{ACWP}$
- ▶ $TCPI = \frac{BAC - BCWP}{EAC - ACWP}$
- ▶ Total program performance efficiency chart.
- ▶ Number of tasks due, completed on time, completed late, and total overdue tasks last month.
- ▶ Cumulative number of tasks planned and completed over time.
- ▶ $\frac{\# \text{ of modified CIs rechecked into CM last month}}{\# \text{ of CIs in CM system}}$ 100
- ▶ $\frac{\# \text{ of new and changed requirements last month}}{\# \text{ of original requirements}}$ 100
- ▶ $\frac{\# \text{ of staff voluntarily leaving last month}}{\# \text{ of staff at beginning of month}}$ 100
- ▶ $\frac{\# \text{ of overtime hours last month}}{\# \text{ of base hours}}$ 100
- ▶ Each risk plotted in regions of high, moderate, and low risk exposure.
- ▶ Risk Reserve Dollars: Total cost risk exposures compared with cost risk reserve.
- ▶ Risk Reserve Time: Total schedule risk exposures compared with schedule risk reserve.
- ▶ Anonymous Channel Warnings—had news from staff.
- ▶ Total # of Severity 1 & 2 defects that are open and closed.

*For More Information see "Project Control Panel," *The Program Manager's Guide to Software Acquisition Best Practices*, Software Program Managers Network, June 1998, Chapter 2.

3. BREATHALYZER TEST

If a program manager cannot answer the following questions about current project status, or must answer in the negative, then the project should be scheduled for immediate review.

1. Do you have a current, credible activity network supported by a Work Breakdown Structure (WBS)?
2. Do you have a current, credible schedule and budget?
3. Do you know what software you are responsible for delivering?
4. Can you list the current top ten project risks?
5. Do you know your schedule compression percentage?
6. What is the estimated size of your software deliverable? How was it derived?
7. Do you know the percentage of external interfaces that are not under your control?
8. Does your staff have sufficient expertise in the project domains?
9. Have you identified adequate staff to allocate to the scheduled tasks at the scheduled time?

4. QUANTITATIVE TARGETS

Quantitative targets apply to key project areas being measured, providing best-in-class objectives for DoD-contracted software projects. The targets and their associated warning levels of possible malpractice follow:

MEASUREMENT	TARGET	WARNING LEVEL
DEFECT REMOVAL EFFICIENCY	> 95%	< 85%
SCHEDULE SLIP OR COST IN EXCESS OF RISK RESERVE	0%	≥ 10%
TOTAL REQUIREMENTS GROWTH (IN FUNCTION POINTS OR EQUIVALENT)	≤ 1% PER MONTH	≥ 50% PER YEAR
TOTAL SOFTWARE PROGRAM DOCUMENTATION	< 1000 WORDS PER FUNCTION POINT	> 2000 WORDS PER FUNCTION POINT
VOLUNTARY STAFF TURNOVER PER YEAR	1-3%	≥ 10%
ORIGINAL DEFECT DENSITY	< 4 PER FUNCTION POINT	> 7 PER FUNCTION POINT

5. PRINCIPAL BEST PRACTICES

The Airlie Software Council identified nine Principal Best Practices observed to be used generally and successfully in industry, and deemed essential for nearly all DoD software development projects.

1. FORMAL RISK MANAGEMENT

The discipline of risk management is vital to the success of any software effort. A formal risk management process requires corporate acceptance of risk as a major consideration for software program management, commitment of program resources, and formal methods for identifying, monitoring, and managing risk.

STATUS CHECKS:

- Has a Risk Officer been appointed?
- Has a risk database been set up?
- Do risk assessments have a clear impact on program plans and decisions?
- Is the frequency and timeliness of risk assessment updates consistent with decision updates during the project?
- Are objective criteria used to identify, evaluate, and manage risks?
- Do information flow patterns and reward criteria within the organization support the identification of risk by all project personnel?

- Is the integrity of information managed and controlled?
- Are risks identified throughout the entire life cycle, not just during the current PM's assignment?
- Is there a management reserve for risk resolution?
- Is there a risk profile drawn up for each risk, and is the risk's probability of occurrence, consequences, severity, and delay regularly updated?
- Does the risk management plan contain explicit provisions to alert decision makers when a risk becomes imminent?
- Have all project personnel been chartered to be risk identifiers?

2. AGREEMENT ON INTERFACES

To deal with the chronic problem of vague, inaccurate, and untestable specifications, the Council proposed that a baseline user interface must be agreed upon before the beginning of implementation activities, and that such user interface must be made and maintained as an integral part of the system specification. For those projects developing both hardware and software, a separate software specification must be written with an explicit and complete interface description.

5. PRINCIPAL BEST PRACTICES (CONT.)

STATUS CHECKS:

- Is there a complete census of inputs/outputs? Are such inputs/outputs defined down to the data-element level?
- Are the interfaces stable?
- Have you considered hardware/software, users, major software component interfaces, etc.?
- Have existing and future interfaces been defined, including consideration of those that may be required over time?
- Does the system specification include a separate software specification to show the hardware interfaces?
- Are opportunities made available for users to provide input and review the user interface descriptions as they develop?

3. FORMAL INSPECTIONS

Inspections should be conducted on requirements, architecture, designs at all levels (particularly detailed design), code prior to unit test, and test plans.

STATUS CHECKS:

- Are inspections identified and implemented to assess the quality of all baselined artifacts and placed under control before they are released for project use?

- Is the conduct of inspections structured, and are they integrated into the project schedule?
- Are procedures, standards, and rules for the conduct of the inspections established?
- Are metrics used to gauge the effectiveness of inspections?
- Is there a documented process for conducting inspections?
- Are entrance and exit criteria established for each inspection?
- Are a significant number of defects caught as early as possible (prior to testing at minimum)?
- Are inspections specifically focused on a narrow set of objectives, and do they evaluate a fixed set of data?
- Is there a clear rationale for the scheduling of inspections?
- Are defects from inspections tracked and catalogued?
- Are inspections conducted to assess the quality of all engineering data products before they are released for project use?
- Is the detailed design reviewable?

5. PRINCIPAL BEST PRACTICES (CONT.)

4. METRICS-BASED SCHEDULING AND MANAGEMENT

Statistical quality control of costs and schedules should be maintained. This requires early calculation of size metrics, projection of costs and schedules from empirical patterns, and tracking of project status through the use of captured result metrics. Use of a parametric analyzer or other automated projection tool is also recommended.

STATUS CHECKS:

- Are cost and schedule performance tracked against the initial baseline and the latest baseline?
- Are the number of changes to the initial cost/schedule baseline tracked?
- Does the plan identify progress measures to permit rate charting and tracking?
- Are inspection coverage and error removal rates tracked for the entire product and for each component?
- Are project estimates continuously refined as the project proceeds?
- Is a project feedback loop established between project measures and updated schedules?
- Is there a process for capturing the primitive data necessary to calculate the Earned Value?
- Are productivity levels and schedule deadlines

evaluated against past performance and reflected in the risk assessment?

- Are the planned vs. actual cost and planned vs. actual schedule monitored?
- Is there automated support for metrics-based scheduling and tracking procedures?

5. BINARY QUALITY GATES AT THE INCH-PEBBLE LEVEL *

Completion of each task in the lowest-level activity network needs to be defined by an objective binary indication. These completion events should be in the form of gates that assess either the quality of the products produced or the adequacy and completeness of the finished process. Gates may take the form of technical reviews, completion of a specific set of tests which integrate or qualify software components, demonstrations, or project audits. The binary indication is meeting a predefined performance standard (e.g., defect density of less than four per Function Point). Activities are closed only upon satisfying the standard, with no partial credit given. Quality gates can be applied at any time during the project—including solicitation.

** The Aerospace Industries Association (AIA) has commented that the use of technical reviews, tests, demonstrations, or audits as "completion criteria for 'inch-pebbles' is excessive in terms of value added to the customer or the contractor." AIA recommended the "retention of the 'Binary Quality Gates' concept and deletion of references to the granularity of the tasks and to the term 'inch-pebbles'."*

5. PRINCIPAL BEST PRACTICES (CONT.)

STATUS CHECKS:

- Have credible project status and planning estimates been produced based on inch-pebble quality gates which can be aggregated at any desirable level?
- Have all activities been decomposed into inch-pebbles?
- Has all near-term work been decomposed into tasks no longer than two weeks in duration?
- Have achievable accomplishment criteria been identified for each task? Are tasks based on overall quality goals and criteria for the project?
- Are quality gates rigorously applied for determining task accomplishment, without exception?
- Is there clear evidence that planned tasks are 100 percent complete before acceptance?
- Is there clear evidence of successful completion of reviews?
- Are inch-pebble tasks on the critical path defined, enabling more accurate assessment of schedule risks and contingency plans?
- Is the set of binary quality gates compatible with the WBS?

6. PROGRAM-WIDE VISIBILITY OF PROGRESS VS. PLAN

The core indicators of project health or dysfunction—the Control Panel indicators—should be made readily available to all project participants. Anonymous channel feedback should be encouraged to enable bad news to move up and down the project hierarchy.

STATUS CHECKS:

- Are status indicators on the Control Panel updated at least monthly?
- Are the status indicators integrated into the management decision process?
- Is basic project status known by all project personnel?
- Can staff report problems as well as successes?
- Are project goals, plans, schedules, and risks available to the project team and interested parties?
- Is anonymous channel feedback visible to all project members?

5. PRINCIPAL BEST PRACTICES (CONT.)

7. DEFECT TRACKING AGAINST QUALITY TARGETS

Defects should be tracked formally at each project phase or activity. Configuration Management (CM) enables each defect to be recorded and traced through to removal. In this approach there is no such thing as a private defect, that is, one detected and removed without being recorded. Initial quality targets (expressed, for example, in defects per Function Point), as well as calculations of remaining or latent defects, are compared to counts of defects removed in order to track progress during testing activities.

STATUS CHECKS:

- Are defect targets established for the project? Are the targets firm?
- Are consequences defined if a product fails to meet the targets?
- Do project quality targets apply to all products?
- Are there circumstances defined under which quality targets are subject to revision?
- What techniques are used to project latent defect counts?
- How are current projected levels of defect removal empirically confirmed as adequate to achieve planned quality targets?

- Is test coverage sufficient to indicate that the latent defect level achieved by the end of testing will be lower than the established quality targets?
- Are the inspection and test techniques employed during the program effective in meeting quality targets?
- Do all discovered defects undergo CM, and are accurate counts achieved for defects discovered and defects removed?
- Is there a closed-loop system linking defect actions from when defects are first detected to when they're resolved?
- Is the defect information defined at a level of granularity that supports an objective assessment of resolution on a periodic basis?

8. CONFIGURATION MANAGEMENT

The discipline of CM is vital to the success of any software effort. CM is an integrated process for identifying, documenting, monitoring, evaluating, controlling, and approving all changes made during the life cycle of the program for information that is shared by more than one individual or organization.

5. PRINCIPAL BEST PRACTICES (CONT.)

STATUS CHECKS:

- Is the CM process integrated with the project plan and an integral part of the culture?
- Are all versions controlled?
- Are configuration control tools used for status accounting and configuration identification tracking?
- Are periodical reviews and audits in place to assess the effectiveness of the CM process?
- Are all pieces of information shared by two or more organizations placed under CM?
- Do you have a process to measure the cycle time?

9. PEOPLE-AWARE MANAGEMENT ACCOUNTABILITY

Management must be accountable for staffing qualified people (those with domain knowledge and similar experience in previously successful projects), as well as for fostering an environment conducive to high morale and low voluntary staff turnover.

STATUS CHECKS:

- Will the procuring/developing program manager be on board for the entire project?
- Are domain experts available?

- Does the project manager have software experience in a project of similar size and objective?
- Are all personnel fully aware of their roles in the project?
- Is quality of performance acknowledged?
- Is personnel continuity ensured in light of changing company or program needs?
- Are opportunities for professional growth available to all members of the project?
- Do the developers believe in the goals of the project and that the schedule is feasible?
- Is the motivation and retention of personnel a key part of management assessment?

6. BEST PRACTICES

The Software Acquisition Best Practices are derived from practices used by successful commercial and defense software projects.

Because the Best Practices are not tied to a specific metric or method, program managers can selectively apply them to particular corporate and program needs as appropriate.

The Best Practices that follow are discussed in *The Program Manager's Guide to Software Acquisition Best Practices*.

RISK MANAGEMENT

- Establish Management Reserves for Risk Resolution
- Implement Metrics-Based Risk Decisions
- Perform Continuous Risk Management
- Formalize Risk Tracking and Review
- Manage Impact of External Dependencies

PLANNING

- Quantitative Software Estimation/Verification
- Joint Team Involvement
- Activity Planning
- Data Requirements

PROGRAM VISIBILITY

- Practical Project-Oriented Software Measurement Process
- Issue-Driven Measures
- Internal Engineering Analysis Process
- Effective Communication Structure

PROGRAM CONTROL

- Test Methodology
- Regression Testing
- Computer-Aided Software Testing
- Error Source Location
- IV&V
- Quality Gate Completion Criteria
- Configuration Management Coverage
- Requirements Change Management
- Baseline Methodology
- Technical Quality Assurance

6. BEST PRACTICES (CONT.)

ENGINEERING BEST PRACTICES AND CULTURE

- Include User in a Multidisciplined Requirements Support Team
- Encourage Compatible Analysis and Design Methods
- Encourage Software Architecture Definition and Maintenance
- Encourage Requirements Engineering Process that Includes Use of Prototypes, Models, and Simulations
- Encourage Proactive Change Impact Analysis
- Plan for Domain Engineering in Acquisitions
- Encourage Use of Clean Room Techniques
- Enterprise Practices Tailored to Projects
- Encourage Use of Software Development Standards such as MIL-STD-498
- Assess Organizational Effectiveness

PROCESS IMPROVEMENT BEST PRACTICES

- Identify and Foster Sponsorship
- Establish and Maintain the Framework for Process Improvement

- Assess and Reassess an Organization's Process Capability
- Develop a Software Process Improvement Plan
- Institutionalize the Software Process Improvement Plan
- Close the Loop for Software Process Improvement

SOLICITATION AND CONTRACTING

- Management of COTS, Reuse, and Emerging Technologies
- Employ a Customer/Contractor IPT
- Use of Periodic Demos
- Utilize Software Development Capability Evaluations (SDCEs)

7. PROJECT CAVEATS

The following software management caveats are lessons learned from software and hardware/software projects gone awry:

1. Don't expect schedule compression of 10 percent or more compared to the statistical norm for similar projects.
2. Don't justify new technology by the need for schedule compression.
3. Don't force customer-specific implementation solutions on the program.
4. Don't advocate use of silver bullet approaches.
5. Don't miss an opportunity to move items that are under external control off the critical path.
6. Don't bury all project complexity in software as opposed to hardware.
7. Don't conduct critical system engineering tasks without sufficient software engineering expertise.
8. Don't expect to achieve an accurate view of project health from a formal review attended by a large number of unprepared, active reviewers.
9. Don't expect to recover from a schedule slip of 10 percent or more without a 10 percent or greater reduction in software functionality to be delivered.

THE AIRLIE SOFTWARE COUNCIL

Victor Basili	University of Maryland
Grady Booch	Rational
Norm Brown	Software Program Managers Network
Peter Chen	Chen & Associates, Inc.
Christine Davis	Texas Instruments
Tom DeMarco	The Atlantic Systems Guild
Mike Dyer	Lockheed Martin
Mike Evans	Computers & Concepts Associates
Bill Hetzel	Qware
Capers Jones	Software Productivity Research, Inc.
Tim Lister	The Atlantic Systems Guild
John Manzo	3Com
Lou Mazzuchelli	Gerard Klauer Mattison & Co., Inc.
Tom McCabe	McCabe & Associates
Frank McGrath	Software Focus, Inc.
Roger Pressman	R.S. Pressman & Associates, Inc.
Larry Putnam	Quantitative Software Management
Howard Rubin	Hunter College, CUNY
Ed Yourdon	American Programmer